Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 01

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Array.

```
#include<iostream.h>
#include<conio.h>
void main()
{
   int A[100];
   int n, sum = 0;
   clrscr();
   cout << "Enter size of array: ";
   cin >> n;
   for (int i = 0; i < n; i++) {
      cout << "Enter array element: ";
      cin >> A[i];
   }
   for (int j = 0; j < n; j++) {
      sum += A[j];
   }
   cout << "Summation of Array is = " << sum << endl;
   getch();
}
```

Output:

Enter size of array : 5

Enter Array elements:

10

20

30

40

50

Summation of Array is = 150

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 02

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Multidimensional Arrays / Matrices

```
#include<iostream.h>
#include<conio.h>
void main()
{
  int rows, cols,i,j;
  int arr1[10][10], arr2[10][10], sum[10][10];
  clrscr();
  // Input the dimensions of the matrix
  cout << "Enter the number of rows and columns: ";
  cin >> rows >> cols;
  // Input elements of the first matrix
  cout << "Enter elements of first matrix: \n";
  for ( i = 0; i < rows; i++) {
    for ( j = 0; j < cols; j++) {
      cin >> arr1[i][j];
    }
  }
  // Input elements of the second matrix
  cout << "Enter elements of second matrix: \n";
  for ( i = 0; i < rows; i++) {
    for ( j = 0; j < cols; j++) {
```

```
            cin >> arr2[i][j];
        }
    }
    // Adding the two matrices
    for ( i = 0; i < rows; i++) {
        for ( j = 0; j < cols; j++) {
            sum[i][j] = arr1[i][j] + arr2[i][j];
        }
    }
    // Display the sum matrix
    cout << "Sum of the matrices: \n";
    for ( i = 0; i < rows; i++) {
        for ( j = 0; j < cols; j++) {
            cout << sum[i][j] << " ";
        }
        cout << endl;
    }
    getch();
}
```

Output:

Enter the number of rows and columns: 2 2

Enter elements of first matrix: 1 2   2 1

Enter elements of second matrix: 2 2   1 1

Sum of the matrices: 3 4   3 2

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 03

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Stack .

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
int n;
class stack
{
private:
int s[10],top,ele,i;  // char s[10] for character
public:
stack()
{
top=-1;
}
void push();
void dis();
void pop();
void peep();
void change();
};
void stack::push()
{
```

```
if(top>=n-1)
cout<<"\nStack is overflow:";
else
{
cout<<"\nEnter element:";
cin>>ele;
top++;
s[top]=ele;
}
}
void stack::dis()
{
if(top==-1)
{
cout<<"\n Stack is Empty";
}
else
{
cout<<"\nElements in stack are:\n";
for(i=top;i>=0;i--)
cout<<s[i]<<"\t";
}
}
void stack::pop()
{
if(top==-1)
```

```
{
cout<<"\nUnderflow";
}
else
{
cout<<"\nPop ele is "<<s[top];
top--;
}
}
void stack::peep()
{
cout<<"\nEnter position:";
cin>>i;
if((top-i+1)<0)
{
cout<<"\nUnderflow";
}
else
{
cout<<"\nPeep ele is "<<s[top-i+1];
}
}
void stack::change()
{
cout<<"\nEnter position ";
cin>>i;
```

```
if((top-i+1)<0)
{
cout<<"\nUnderflow";
}
else
{
int n;              //char n; for character
cout<<"\nEnter element:";
cin>>n;
s[top-i+1]=n;
}}
void main()
{
clrscr();
stack s;
cout<<"Enter size of stack";
cin>>n;
int ch;
cout<<"\n1. Push  2.Display  3.Pop  4.Peep  5.Change 6.Exit\n";
while(ch!=6)
{
cout<<"\nEnter ch :";
cin>>ch;
switch(ch)
{
case 1: s.push(); break;
```

```
case 2: s.dis(); break;

case 3: s.pop();break;

case 4: s.peep(); break;

case 5: s.change(); break;

case 6: exit(0);

}}

getch();

}

*/ Output */

Enter size of stack  3

1. Push  2.Display  3.Pop  4.Peep  5.Change 6.Exit

Enter ch :1

Enter element:10

Enter ch :1

Enter element:20

Enter ch :1

Enter element:30

Enter ch :1

Stack is overflow:

Enter ch :2

Elements in stack are: 30      20      10

Enter ch :3

Pop ele is 30

Enter ch :2

Elements in stack are: 20      10

Enter ch :4
```

Enter position:1

Peep ele is 20

Enter ch : 2

Elements in stack are: 20 10

Enter ch :5

Enter position 1

Enter element:80

Enter ch :2

Elements in stack are: 80 10

Enter ch : 6

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 04

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Infix to Postfix .

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
class Stack {
private:
   char st[100];
   int top;
public:
   Stack() { top = -1; }
   void push(char c) { st[++top] = c; }
   char pop() { return st[top--]; }
   char peek() { return st[top]; }
   int empty() { return top == -1; }
};
int precedence(char op) {
   if (op == '^') return 3;
   if (op == '*' || op == '/') return 2;
   if (op == '+' || op == '-') return 1;
   return 0;
}
```

```
int isOperator(char c) {
   return (c == '+' || c == '-' || c == '*' || c == '/' || c == '^');
}
int isalnum(char c) {
   return ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') || (c >= '0' && c <= '9'));
}
void infixToPostfix(char infix[], char postfix[]) {
   Stack s;
   int i = 0, j = 0;
   while (infix[i]) {
      char ch = infix[i];
      if (isalnum(ch)) {
         postfix[j++] = ch;
      }
      else if (ch == '(') {
         s.push(ch);
      }
      else if (ch == ')') {
         while (!s.empty() && s.peek() != '(') {
            postfix[j++] = s.pop();
         }
         s.pop(); // Remove '('
      }
      else if (isOperator(ch)) {
         while (!s.empty() && precedence(s.peek()) >= precedence(ch)) {
            postfix[j++] = s.pop();
```

```
        }
        s.push(ch);
      }
      i++;
   }
   while (!s.empty()) {
      postfix[j++] = s.pop();
   }
   postfix[j] = '\0';
}
void main() {
   char infix[100], postfix[100];
   clrscr();

   cout << "Enter an infix expression: ";
   cin >> infix;
   infixToPostfix(infix, postfix);
   cout << "Postfix Expression: " << postfix << endl;
   getch();
}
```

Output:

Enter an infix expression: +ABC

Postfix Expression: ABC+

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 05

Date of Performance: Date of Completion:

Name : Roll No :

Title : To implement a program in cpp for Infix to Prefix.

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
class convert
{
char infix[20],postfix[20],s[20];
int i,p,top;
public:
convert()
{
top=-1;
i=p=0;
cout<<"\nEnter infix Expression:";
cin>>infix;
strcat(infix,")");
s[++top]='(';
}
int precedance(char);
void post();
void display();
};
```

```
int convert::precedance(char ch)
{
switch(ch)
{
case '^':return 3;
case '*':return 2;
case '/':return 2;
case '+':return 1;
case '-':return 1;
default: return 0;
}}
void convert::post()
{
char ch;
while(top!=-1)
{
ch=infix[i++];
if((ch>='A'&&ch<='Z')||(ch>='a'&&ch<='z')||(ch>='1'&&ch<='9'))
postfix[p++]=ch;
else if(ch=='(')
s[++top]=ch;
else if(ch=='+'||ch=='-'||ch=='*'||ch=='/'||ch=='^')
{
while(precedance(ch)<=precedance(s[top]))
postfix[p++]=s[top--];
s[++top]=ch;
```

```
}
else if(ch==')')
{
while(s[top]!='(')
postfix[p++]=s[top--];
top--;
}
else
cout<<"\nWrong string";
}
postfix[p]='\0';
}
void convert::display()
{
cout<<"\nPostfix Expression is :"<<postfix;
}
void main()
{
clrscr();
convert c;
c.post();
c.display();
getch();
}
*/ Output */
Enter infix Expression:(a*b-(c+d/e^f)*h)
```

Postfix Expression is :ab*cdef^/+h*-

Enter infix Expression:a+2*5

Postfix Expression is :a25*

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 06

Date of Performance: Date of Completion:

Name : Roll No :

Title:To implement a program in cpp for Queue using array.

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
int m;  // Size of the queue
class queue {
   int f, r, q[10], n;  // f = front, r = rear, q = array for queue, n = element to insert
public:
   // Constructor to initialize front and rear to 0
   queue() {
      f = r = 0;
   }
   void insert();  // Insert function
   void del();     // Delete function
   void dis();     // Display function
};
// Function to insert an element in the queue
void queue::insert() {
   if (r == m) {
      cout << "\nOverflow (Queue is full)";
   } else {
      cout << "\nEnter Element in Queue = ";
```

```
        cin >> n;  // Read the element to be inserted
        if (f == 0) {
            f = 1;  // Set front to 1 if it's the first insertion
        }
        r++;  // Move the rear to the next position
        q[r] = n;  // Insert the element at the rear
    }
}
// Function to delete an element from the queue
void queue::del() {
    if (f == 0) {
        cout << "\nUnderflow (Queue is empty)";
    } else {
        int n = q[f];  // Get the element at the front
        if (f == r) {
            // If front equals rear, the queue becomes empty after deletion
            f = r = 0;
        } else {
            f++;  // Move the front to the next position
        }
        cout << "\nDeleted element is " << n;
    }
}
// Function to display the elements in the queue
void queue::dis() {
    if (f == 0) {
```

```
        cout << "\nUnderflow (Queue is empty)";
    } else {
        cout << "\nElements in queue are: ";
        for (int i = f; i <= r; i++) {
            cout << q[i] << "\t";  // Display each element from front to rear
        }
    }
}
void main() {
    queue q;  // Create an object of the queue class
    int ch;   // Variable to store user's choice
    clrscr();
    cout << "Enter size of queue: ";
    cin >> m;  // Read the maximum size of the queue
    cout << "\n1. Insert \n2. Display \n3. Delete \n4. Exit \n";
    while (1) {
        cout << "\nEnter choice: ";
        cin >> ch;  // Read user's choice

        switch (ch) {
            case 1:
                q.insert();  // Call insert function
                break;
            case 2:
                q.dis();     // Call display function
                break;
```

```
        case 3:
            q.del();    // Call delete function
            break;
        case 4:
            getch();
            exit(0);    // Exit the program
        default:
            cout << "\nInvalid choice!";
      }
   }
}
```

Output:

```
Enter size of queue: 5
1. Insert
2. Display
3. Delete
4. Exit
Enter choice: 1
Enter Element in Queue = 1
Enter choice: 1
Enter Element in Queue = 2
Enter choice: 1
Enter Element in Queue = 3
Enter choice: 1
```

Enter Element in Queue = 4

Enter choice: 1

Enter Element in Queue = 5

Enter choice: 2

Elements in queue are: 1 2 3 4 5

Enter choice: 3

Deleted element is 1

Enter choice: 4

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 08

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in CPP for Deques.

```
#include<iostream.h>
#include<conio.h>
#define SIZE 5
int deque[SIZE];
int f = -1, r = -1;
// insert_front function will insert the value from the front
void insert_front(int x) {
   if ((f == 0 && r == SIZE - 1) || (f == r + 1)) {
      cout << "Overflow" << endl;
   } else if (f == -1 && r == -1) {
      f = r = 0;
      deque[f] = x;
   } else if (f == 0) {
      f = SIZE - 1;
      deque[f] = x;
   } else {
      f = f - 1;
      deque[f] = x;
   }
}
// insert_rear function will insert the value from the rear
```

```
void insert_rear(int x) {
   if ((f == 0 && r == SIZE - 1) || (f == r + 1)) {
      cout << "Overflow" << endl;
   } else if (f == -1 && r == -1) {
      r = 0;
      deque[r] = x;
   } else if (r == SIZE - 1) {
      r = 0;
      deque[r] = x;
   } else {
      r++;
      deque[r] = x;
   }
}
// display function prints all the values of deque
void display() {
   if (f == -1 && r == -1) {
      cout << "Deque is empty" << endl;
      return;
   }

   int i = f;
   cout << "Elements in deque are: ";
   do {
      cout << deque[i] << " ";
      i = (i + 1) % SIZE;
```

```
    } while (i != (r + 1) % SIZE);
    cout << endl;
}
// getfront function retrieves the first value of the deque
void getfront() {
    if (f == -1 && r == -1) {
        cout << "Deque is empty" << endl;
    } else {
        cout << "The value at the front is: " << deque[f] << endl;
    }
}
// getrear function retrieves the last value of the deque
void getrear() {
    if (f == -1 && r == -1) {
        cout << "Deque is empty" << endl;
    } else {
        cout << "The value at the rear is: " << deque[r] << endl;
    }
}
// delete_front() function deletes the element from the front
void delete_front() {
    if (f == -1 && r == -1) {
        cout << "Deque is empty" << endl;
    } else if (f == r) {
        cout << "The deleted element is: " << deque[f] << endl;
        f = r = -1;
```

```
   } else if (f == SIZE - 1) {
      cout << "The deleted element is: " << deque[f] << endl;
      f = 0;
   } else {
      cout << "The deleted element is: " << deque[f] << endl;
      f = f + 1;
   }
}
// delete_rear() function deletes the element from the rear
void delete_rear() {
   if (f == -1 && r == -1) {
      cout << "Deque is empty" << endl;
   } else if (f == r) {
      cout << "The deleted element is: " << deque[r] << endl;
      f = r = -1;
   } else if (r == 0) {
      cout << "The deleted element is: " << deque[r] << endl;
      r = SIZE - 1;
   } else {
      cout << "The deleted element is: " << deque[r] << endl;
      r = r - 1;
   }
}
void main() {
   clrscr();
   insert_front(20);
```

```
    insert_front(10);
    insert_rear(30);
    insert_rear(50);
    insert_rear(80);
    display(); // Display the values of deque
    getfront(); // Retrieve the value at front-end
    getrear();  // Retrieve the value at rear-end
    delete_front();
    delete_rear();
    display(); // Display the values after deletion
    getch();
}
```

Output:

```
Elements in deque are: 10 20 30 50 80
The value at the front is: 10
The value at the rear is: 80
The deleted element is: 10
The deleted element is: 80
Elements in deque are: 20 30 50
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 08

Date of Performance: Date of Completion:

Name : Roll No :

Title : To implement a program in CPP for Linear(Single) Linked List.

```
#include<iostream.h>
#include<conio.h>
#include<process.h>

class node {
   int info, item, s;
   node *link;
public:
   void insert();
   void dis();
   void del();
   void search();
   void sum();
};

node *move, *start = NULL, *temp;

void node::insert() {
   cout << "\nEnter the item:";
   cin >> item;
   node *node1 = new node;
```

```
    node1->link = NULL;
    node1->info = item;

    if (start == NULL)
        start = node1;
    else {
        move = start;
        while (move->link != NULL)
            move = move->link;
        move->link = node1;
    }
}

void node::dis() {
    node *x;
    x = start;

    cout << "\nElements in LL are:";
    while (x != NULL) {
        cout << "\t" << x->info;
        x = x->link;
    }
}

void node::sum() {
    node *x;
```

```
    x = start;
    s = 0;
    while (x != NULL) {
        s = s + x->info;
        x = x->link;
    }
    cout << "\nSum of node is " << s;
}

void node::del() {
    temp = start;
    if (temp != NULL) {
        temp = temp->link;
        cout << "\nDeleted node is " << start->info;
        delete start;
        start = temp;
    } else
        cout << "\nList is empty:";
}

void node::search() {
    int c = 0, f = 0, d;
    cout << "\nEnter item: ";
    cin >> item;
    temp = start;
```

```
    while (temp != NULL) {
        c++;
        if (temp->info == item) {
            f = 1;
            d = c;
            break;
        }
        temp = temp->link;
    }

    if (f == 1)
        cout << "\nElement is found at position " << d;
    else
        cout << "\nElement is not found";
}

void main() {
    node n;
    int ch;
    clrscr();

    cout << "\n1.Insert 2.Display 3.Delete 4.Search 5.Sum 6.Exit \n";

    do {
        cout << "\nEnter choice: ";
        cin >> ch;
```

```
      switch (ch) {
         case 1: n.insert(); break;
         case 2: n.dis(); break;
         case 3: n.del(); break;
         case 4: n.search(); break;
         case 5: n.sum(); break;
         case 6: getch(); exit(0);
      }
   } while (ch != 6);
}
```

Output:

1.Insert 2.Display 3. Delete 4.Search 5.Sum 6.Exit

Enter choice1

Enter the item:1

Enter choice1

Enter the item:2

Enter choice1

Enter the item:3

Enter choice1

Enter the item:4

Enter choice1

Enter the item:5

Enter choice2

Elements in LL are: 1 2 3 4 5

Enter choice3

Deleted node is1

Enter choice4

Enter item2

Element is found at position 1

Enter choice5

Sum of node is14

Enter choice

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 9

Date of Performance: Date of Completion:

Name : Roll No :

Title : To implement a program in CPP for Circular Linked List.

```
 #include<iostream.h>
#include<conio.h>
#include<process.h>

class node {
   int info, c, i;
   node *link;
public:
   node() {
      c = 0;
   }
   void insert();
   void display();
   void del();
};

node *start = NULL, *temp = NULL, *move = NULL, *temp1 = NULL;

void node::insert() {
   int item;
   node *p = new node;
```

```
    cout << "\nEnter Element:";
    cin >> item;

    p->info = item;
    p->link = NULL;

    if (start == NULL) {
        start = p;
        p->link = start;
        c++;
    } else {
        temp = start;
        while (temp->link != start)
            temp = temp->link;
        temp->link = p;
        p->link = start;
        c++;
    }
}

void node::display() {
    if (start == NULL) {
        cout << "\nLL empty";
        return;
    }
```

```
    node *temp;
    temp = start;
    move = start->link;
    cout << temp->info;
    while (move != start) {
        cout << "->" << move->info;
        move = move->link;
    }
    cout << "\nNumber of nodes in CLL are: " << c;
}

void node::del() {
    int pos;
    cout << "\nEnter Position:";
    cin >> pos;

    if (c == 1) {
        delete start;
        start = NULL;
        c--;
        return;
    }
     if (start == NULL) {
        cout << "\nLL Empty:";
        return;
    }
```

```
if (pos > c || pos < 1) {
   cout << "\nInvalid Position";
   return;
}
   if (pos == 1) {
   temp = start;
   while (temp->link != start)
      temp = temp->link;
   temp1 = start;
   start = start->link;
   temp->link = start;
   cout << "\nDeleted Element is " << temp1->info;
   delete temp1;
   c--;
} else {
   temp = start;
   i = 1;
   while (i < pos - 1) {
      temp = temp->link;
      i++;
   }
   temp1 = temp->link;
   temp->link = temp1->link;
   cout << "\nDeleted element is " << temp1->info;
   delete temp1;
   c--;
```

```
    }
}
void main() {
    node n;
    int ch;
    clrscr();
    cout << "\n1.Insert 2.Display 3.Delete 4.Exit";
        while (1) {
        cout << "\nEnter Choice: ";
        cin >> ch;
        switch (ch) {
            case 1: n.insert(); break;
            case 2: n.display(); break;
            case 3: n.del(); break;
            case 4: getch(); exit(0);
        }
    }
}
```

Output:

1. Insert
2. Display
3. Delete
4. Exit

Enter Choice: 1

Enter Element:

10

Enter Choice: 1

Enter Element:

20

Enter Choice: 1

Enter Element:

30

Enter Choice: 2

10->20->30

Number of

nodes in CLL

are: 3

Enter Choice: 3

Enter Position:

2

Deleted

Element is 20

Enter Choice: 2

10->30

Number of

nodes in CLL

are: 2

Enter Choice: 4

Exit

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 10

Date of Performance: Date of Completion:

Name : Roll No :

Title : To implement a program in CPP for Doubly Linked List.

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
class node {
   int info, c, j;
   node *left, *right;
public:
   void insert();
   void display();
   void del();
};
node *start = NULL, *temp = NULL, *move = NULL, *temp1 = NULL;
void node::insert() {
   int item;
   node *p = new node;
   cout << "\nEnter element:";
   cin >> item;
   p->info = item;
   p->left = NULL;
   p->right = NULL;
   if (start == NULL) {
```

```
        start = p;
        return;
    } else {
        temp = start;
        while (temp->right != NULL)
            temp = temp->right;
        temp->right = p;
        p->left = start;
    }
}
void node::display() {
    move = start;
    if (move == NULL) {
        cout << "\nLL Empty:";
        return;
    } else {
        cout << "\nnode in DLL are :";
        while (move != NULL) {
            cout << move->info << "\t";
            move = move->right;
        }
    }
}
void node::del() {
    if (start == NULL) {
        cout << "\nLL Empty:";
```

```
        return;
    }
    temp = start;
    start = temp->right;
    if (start != NULL)
        start->left = NULL;
    temp->right = NULL;

    cout << "\ndeleted element is " << temp->info;
    delete temp;
}

void main() {
    node n;
    int ch;
    clrscr();
    cout << "\n1. Insert 2. Display 3. Delete 4. Exit";
    while (1) {
        cout << "\nEnter choice: ";
        cin >> ch;
        switch (ch) {
            case 1: n.insert(); break;
            case 2: n.display(); break;
            case 3: n.del(); break;
            case 4: getch(); exit(0);
        }
```

```
   }
}
```

Output :

```
1. Insert 2. Display 3. Delete 4. Exit
Enter choice: 1
Enter element: 10
Enter choice: 1
Enter element: 20
Enter choice: 1
Enter element: 30
Enter choice: 2
Nodes in DLL are: 10    20    30
Enter choice: 3
Deleted element is: 10
Enter choice: 2
Nodes in DLL are: 20    30
Enter choice: 4
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 11

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement the program in CPP for Polyno mial Addition.

```
#include<iostream.h>
#include<conio.h>
void main() {
   int a[10], b[10], c[10];
   int m, n, k = 0, k1, i, j;
   clrscr();
   cout << "\n\tPolynomial Addition \n";
   cout << "\t================== \n";
   // Input for the first polynomial
   cout << "\n\tEnter the number of terms of the polynomial: ";
   cin >> m;
   cout << "\n\tEnter the degrees and coefficients: ";
   for (i = 0; i < 2 * m; i++) {
      cin >> a[i];
   }

   // Display the first polynomial
   cout << "\n\tFirst polynomial is: ";
   k1 = 0;
   if (a[k1 + 1] == 1) {
      cout << "x^" << a[k1];
```

```
} else {
   cout << a[k1 + 1] << "x^" << a[k1];
}
k1 += 2;
while (k1 < 2 * m) {
   cout << "+" << a[k1 + 1] << "x^" << a[k1];
   k1 += 2;
}

// Input for the second polynomial
cout << "\n\n\tEnter the number of terms of the second polynomial: ";
cin >> n;
cout << "\n\tEnter the degrees and coefficients: ";
for (j = 0; j < 2 * n; j++) {
   cin >> b[j];
}

// Display the second polynomial
cout << "\n\tSecond polynomial is: ";
k1 = 0;
if (b[k1 + 1] == 1) {
   cout << "x^" << b[k1];
} else {
   cout << b[k1 + 1] << "x^" << b[k1];
}
k1 += 2;
```

```
while (k1 < 2 * n) {
   cout << "+" << b[k1 + 1] << "x^" << b[k1];
   k1 += 2;
}

// Add the two polynomials
i = 0; j = 0;
while (m > 0 && n > 0) {
   if (a[i] == b[j]) {
      c[k + 1] = a[i + 1] + b[j + 1];
      c[k] = a[i];
      m--; n--;
      i += 2; j += 2;
   } else if (a[i] > b[j]) {
      c[k + 1] = a[i + 1];
      c[k] = a[i];
      m--;
      i += 2;
   } else {
      c[k + 1] = b[j + 1];
      c[k] = b[j];
      n--;
      j += 2;
   }
   k += 2;
}
```

```
// Add remaining terms of the first polynomial
while (m > 0) {
   c[k + 1] = a[i + 1];
   c[k] = a[i];
   k += 2; i += 2; m--;
}

// Add remaining terms of the second polynomial
while (n > 0) {
   c[k + 1] = b[j + 1];
   c[k] = b[j];
   k += 2; j += 2; n--;
}

// Display the resulting polynomial
cout << "\n\n\tSum of the two polynomials is: ";
k1 = 0;
if (c[k1 + 1] == 1) {
   cout << "x^" << c[k1];
} else {
   cout << c[k1 + 1] << "x^" << c[k1];
}
k1 += 2;
while (k1 < k) {
   if (c[k1 + 1] == 1) {
```

```
        cout << "+x^" << c[k1];
      } else {
        cout << "+" << c[k1 + 1] << "x^" << c[k1];
      }
      k1 += 2;
    }

    cout << endl;
    getch();
}
```

Output:

```
Enter the number of terms of the polynomial: 2
Enter the degrees and coefficients: 2 3 0 4
First polynomial is: 3x^2+4x^0
Enter the number of terms of the second polynomial: 2
Enter the degrees and coefficients: 3 1 2 2
Second polynomial is: 1x^3+2x^2
Sum of the two polynomials is: 1x^3+5x^2+4x^0
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 12

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement the program in CPP for Polynomial Addition sing Linked List.

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
struct Node {
   int coeff;
   int pow;
   Node* next;
};
void create_node(int coeff, int pow, Node** temp) {
   Node *r, *z;
   z = *temp;
   if (z == NULL) {
      r = new Node();
      r->coeff = coeff;
      r->pow = pow;
      *temp = r;
      r->next = new Node();
      r = r->next;
      r->next = NULL;
   } else {
      r = z;
```

```
        while (r->next != NULL)
            r = r->next;
        r->next = new Node();
        r = r->next;
        r->coeff = coeff;
        r->pow = pow;
        r->next = NULL;
    }
}
void polyadd(Node* poly1, Node* poly2, Node* poly) {
    while (poly1->next != NULL && poly2->next != NULL) {
        if (poly1->pow > poly2->pow) {
            poly->pow = poly1->pow;
            poly->coeff = poly1->coeff;
            poly1 = poly1->next;
        } else if (poly1->pow < poly2->pow) {
            poly->pow = poly2->pow;
            poly->coeff = poly2->coeff;
            poly2 = poly2->next;
        } else {
            poly->pow = poly1->pow;
            poly->coeff = poly1->coeff + poly2->coeff;
            poly1 = poly1->next;
            poly2 = poly2->next;
        }
        poly->next = new Node();
```

```
        poly = poly->next;
        poly->next = NULL;
    }
    while (poly1->next != NULL) {
        poly->pow = poly1->pow;
        poly->coeff = poly1->coeff;
        poly1 = poly1->next;
        poly->next = new Node();
        poly = poly->next;
        poly->next = NULL;
    }
    while (poly2->next != NULL) {
        poly->pow = poly2->pow;
        poly->coeff = poly2->coeff;
        poly2 = poly2->next;
        poly->next = new Node();
        poly = poly->next;
        poly->next = NULL;
    }
}

void show(Node* node) {
    while (node->next != NULL) {
        cout << node->coeff << "x^" << node->pow;
        node = node->next;
        if (node->coeff >= 0 && node->next != NULL)
```

```
        cout << "+";
    }
}

void main() {
    Node *poly1 = NULL, *poly2 = NULL, *poly = NULL;
    clrscr();
     // Create first polynomial: 5x^2 + 4x^1 + 2x^0
    create_node(5, 2, &poly1);
    create_node(4, 1, &poly1);
    create_node(2, 0, &poly1);
       // Create second polynomial: -5x^1 - 5x^0
    create_node(-5, 1, &poly2);
    create_node(-5, 0, &poly2);
       cout << "1st Polynomial: ";
    show(poly1);
    cout << "\n2nd Polynomial: ";
    show(poly2);
    // Create result polynomial
    poly = new Node();
    // Add the two polynomials
    polyadd(poly1, poly2, poly);
       // Display the result
    cout << "\nAdded Polynomial: ";
    show(poly);
       getch();
```

}

Output:

1st Polynomial: 5x^2+4x^1+2x^0

2nd Polynomial: -5x^1 -5x^0

Added Polynomial: 5x^2 -3x^0

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 13

Date of Performance: Date of Completion:

Name : Roll No :

Title:To implement a program in cpp for Binary Search Tree.

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
struct Node {
   int data;
   Node *left_child;
   Node *right_child;
   Node(int value) {
      data = value;
      left_child = NULL;
      right_child = NULL;
   }
};
Node* new_node(int x) {
   return new Node(x);
}
Node* search(Node* root, int x) {
   if (root == NULL || root->data == x)
      return root;
   if (x > root->data)
      return search(root->right_child, x);
```

```
    else
        return search(root->left_child, x);
}
Node* insert(Node* root, int x) {
    if (root == NULL)
        return new_node(x);
    if (x > root->data)
        root->right_child = insert(root->right_child, x);
    else
        root->left_child = insert(root->left_child, x);
    return root;
}
Node* find_minimum(Node* root) {
    if (root == NULL)
        return NULL;
    if (root->left_child != NULL)
        return find_minimum(root->left_child);
    return root;
}
Node* delete_node(Node* root, int x) {
    if (root == NULL)
        return NULL;
    if (x > root->data)
        root->right_child = delete_node(root->right_child, x);
    else if (x < root->data)
        root->left_child = delete_node(root->left_child, x);
```

```
    else {
        // Node with no children
        if (root->left_child == NULL && root->right_child == NULL) {
            delete root;
            return NULL;
        }
        // Node with one child
        else if (root->left_child == NULL || root->right_child == NULL) {
            Node* temp = (root->left_child == NULL) ? root->right_child : root->left_child;
            delete root;
            return temp;
        }
        // Node with two children
        else {
            Node* temp = find_minimum(root->right_child);
            root->data = temp->data;
            root->right_child = delete_node(root->right_child, temp->data);
        }
    }
    return root;
}
void inorder(Node* root) {
    if (root != NULL) {
        inorder(root->left_child);
        cout << root->data << " ";
        inorder(root->right_child);
```

```
    }
}
void main() {
    Node* root = new_node(20);
    clrscr();
    insert(root, 5);
    insert(root, 1);
    insert(root, 15);
    insert(root, 9);
    insert(root, 7);
    insert(root, 12);
    insert(root, 30);
    insert(root, 25);
    insert(root, 40);
    insert(root, 45);
    insert(root, 42);
    cout << "In-order traversal before deletion: ";
    inorder(root);
    cout << endl;
    // Deleting nodes
    root = delete_node(root, 1);
    root = delete_node(root, 40);
    root = delete_node(root, 45);
    root = delete_node(root, 9);
    cout << "In-order traversal after deletion: ";
    inorder(root);
```

```
    cout << endl;
    getch();
}
```

Output:

1 5 7 9 12 15 20 25 30 40 42 45

5 7 12 15 20 25 30 42

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 15

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for In-order, Pre-order, Post-order Traversals.

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
struct ver {
   int data;
   ver *left, *right;
};

class tree {
public:
   ver* create(int, ver*);
   void in(ver*);
   void post(ver*);
   void pre(ver*);
};

ver* tree::create(int c, ver* node) {
   if (node == NULL) {
      node = new ver;
      node->data = c;
      node->left = NULL;
```

```
        node->right = NULL;
        return node;
    } else {
        if (c < node->data)
            node->left = create(c, node->left);
        else
            node->right = create(c, node->right);
        return node;
    }
}

void tree::in(ver* node) {
    if (node) {
        in(node->left);
        cout << node->data << "\t";
        in(node->right);
    }
}

void tree::pre(ver* node) {
    if (node) {
        cout << node->data << "\t";
        pre(node->left);
        pre(node->right);
    }
}
```

```
void tree::post(ver* node) {
   if (node) {
      post(node->left);
      post(node->right);
      cout << node->data << "\t";
   }
}

void main() {
   tree t;
   ver* r = NULL;
   int n, ch;
   clrscr();

   cout << "\n1: Insert 2: Inorder 3: Preorder 4: Postorder 5: Exit : \n";

   while (1) {
      cout << "\nEnter Choice:";
      cin >> ch;
      switch (ch) {
         case 1:
            cout << "\nEnter Node:";
            cin >> n;
            r = t.create(n, r);
            break;
```

```
            case 2:
               cout << "\nInorder Traversal:";
               t.in(r);
               cout << endl;
               break;
            case 3:
               cout << "\nPreorder Traversal:";
               t.pre(r);
               cout << endl;
               break;
            case 4:
               cout << "\nPostorder Traversal:";
               t.post(r);
               cout << endl;
               break;
            case 5:
               getch();
               exit(0);
         }
      }
}
```

Output:

1: Insert 2: Inorder 3: Preorder 4: Postorder 5: Exit :

Enter Choice: 1

Enter Node: 50

Enter Choice: 1

Enter Node: 30

Enter Choice: 1

Enter Node: 70

Enter Choice: 1

Enter Node: 20

Enter Choice: 1

Enter Node: 40

Enter Choice: 1

Enter Node: 60

Enter Choice: 1

Enter Node: 80

Enter Choice: 2

Inorder Traversal:

20 30 40 50 60 70 80

Enter Choice: 3

Preorder Traversal:

50 30 20 40 70 60 80

Enter Choice: 4

Postorder Traversal:

20 40 30 60 80 70 50

Enter Choice: 5

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 15

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Max -Heap Tree.

```
#include<iostream.h>
#include<conio.h>
class MaxHeap {
private:
  int heap[100];
  int size;
  // Function to heapify up after insertion
  void heapifyUp(int index) {
    if (index == 0) return;
    int parent = (index - 1) / 2;
    if (heap[parent] < heap[index]) {
      int temp = heap[parent];
      heap[parent] = heap[index];
      heap[index] = temp;
      heapifyUp(parent);
    }
  }
  // Function to heapify down after deletion
  void heapifyDown(int index) {
    int left = 2 * index + 1;
    int right = 2 * index + 2;
```

```
        int largest = index;
        if (left < size && heap[left] > heap[largest])
            largest = left;
        if (right < size && heap[right] > heap[largest])
            largest = right;
        if (largest != index) {
            int temp = heap[index];
            heap[index] = heap[largest];
            heap[largest] = temp;
            heapifyDown(largest);
        }
    }
public:
    MaxHeap() {
        size = 0;
    }
    // Function to insert a new element into the heap
    void insert(int value) {
        heap[size] = value;
        heapifyUp(size);
        size++;
    }
    // Function to remove and return the maximum element (root)
    int extractMax() {
        if (size == 0) {
            cout << "Heap is empty!" << endl;
```

```
            return -1;
        }
        int maxElement = heap[0];
        heap[0] = heap[size - 1];
        size--;
        heapifyDown(0);
        return maxElement;
    }
    // Function to display the elements of the heap
    void printHeap() {
        for (int i = 0; i < size; i++) {
            cout << heap[i] << " ";
        }
        cout << endl;
    }
};
void main() {
    MaxHeap maxHeap;
    clrscr();
    // Insert elements into the max heap
    maxHeap.insert(10);
    maxHeap.insert(20);
    maxHeap.insert(15);
    maxHeap.insert(30);
    maxHeap.insert(40);
    cout << "Max Heap after insertions: ";
```

```
    maxHeap.printHeap();
    // Extract maximum elements
    cout << "Extracted max: " << maxHeap.extractMax() << endl;
    cout << "Heap after extraction: ";
    maxHeap.printHeap();
    getch();
}
```

Output:

Max Heap after insertions: 40 30 15 10 20

Extracted max: 40

Heap after extraction: 30 20 15 10

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 16

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in c for Min -Heap Tree.

```
#include<iostream.h>
#include<conio.h>
void minHeapify(int arr[], int n, int i) {
   int smallest = i; // Initialize smallest as root
   int left = 2 * i + 1; // left child
   int right = 2 * i + 2; // right child
     // If left child is smaller than root
   if (left < n && arr[left] < arr[smallest])
      smallest = left;
    // If right child is smaller than smallest so far
   if (right < n && arr[right] < arr[smallest])
      smallest = right;
   // If smallest is not root
   if (smallest != i) {
      int temp = arr[i];
      arr[i] = arr[smallest];
      arr[smallest] = temp;
      // Recursively heapify the affected sub-tree
      minHeapify(arr, n, smallest);
   }
}
```

```
void buildMinHeap(int arr[], int n) {
   // Start from the last non-leaf node and heapify all nodes in reverse order
   for (int i = n / 2 - 1; i >= 0; i--) {
      minHeapify(arr, n, i);
   }
}
void printArray(int arr[], int n) {
   for (int i = 0; i < n; i++)
      cout << arr[i] << " ";
   cout << endl;
}
void main() {
   clrscr();
   // Test with a random array
   int randomArray[] = {4, 10, 3, 5, 15};
   int n1 = 5;
   cout << "Original Random Array: ";
   printArray(randomArray, n1);
   // Build heap
   buildMinHeap(randomArray, n1);
   cout << "Min Heap from Random Array: ";
   printArray(randomArray, n1);
   cout << endl;
   // Test with a sorted array
   int sortedArray[] = {8, 6, 5, 4, 2};
   int n2 = 5;
```

```
    cout << "Original Sorted Array: ";
    printArray(sortedArray, n2);
    // Build heap
    buildMinHeap(sortedArray, n2);
    cout << "Min Heap from Sorted Array: ";
    printArray(sortedArray, n2);
    getch();
}
```

Output:

Original Random Array: 4 10 3 5 15

Min Heap from Random Array: 3 5 4 10 15

Original Sorted Array: 8 6 5 4 2

Min Heap from Sorted Array: 2 4 5 8 6

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 17

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Depth First Traversal.

```
#include<iostream.h>
#include<conio.h>
void main() {
  clrscr();
  cout << "=== Program to demonstrate the DFS Traversal on a Graph, in CPP ===== \n\n";
  // Variable declarations
  int cost[10][10] = {0}; // Adjacency matrix, initialized to 0
  int i, j, k, n, e, top = -1, v;
  int stk[10], visit[10] = {0}, visited[10] = {0};

  cout << "Enter the number of vertices in the Graph: ";
  cin >> n;

  cout << "\nEnter the number of edges in the Graph: ";
  cin >> e;

  cout << "\nEnter the start and end vertex of the edges: \n";
  for (k = 1; k <= e; k++) {
    cin >> i >> j;
    cost[i][j] = 1;
    cost[j][i] = 1; // For undirected graph
```

```
}

cout << "\nEnter the initial vertex to start the DFS traversal with: ";
cin >> v;

cout << "\nThe DFS traversal on the given graph is: \n";
cout << v << " ";
visited[v] = 1; // Mark the starting vertex as visited

k = 1;
while (k < n) {
   for (j = n; j >= 1; j--) {
      if (cost[v][j] != 0 && visited[j] != 1 && visit[j] != 1) {
         visit[j] = 1;
         stk[++top] = j; // Push vertex onto the stack
      }
   }

   if (top == -1) {
      break; // If stack is empty, traversal is complete
   }

   v = stk[top--]; // Pop vertex from the stack
   cout << v << " ";
   visit[v] = 0;  // Mark it as removed from the visit stack
   visited[v] = 1; // Mark it as visited
```

```
    k++;
  }


  cout << "\n";
  getch();
}
```

Output:

```
===== Program to demonstrate the DFS Traversal on a Graph, in CPP =====
Enter the number of vertices in the Graph: 5
Enter the number of edges in the Graph: 4
Enter the start and end vertex of the edges:
1 2
1 3
2 4
3 5
Enter the initial vertex to start the DFS traversal with: 1
The DFS traversal on the given graph is:
1 3 5 2 4
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 18

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Breadth First Traversal.

```
#include<iostream.h>
#include<conio.h>
int n, i, j, visited[10], queue[10], front = -1, rear = -1;
int adj[10][10];
void bfs(int v) {
  // Mark the starting node as visited
  visited[v] = 1;
  queue[++rear] = v; // Enqueue the starting vertex
  while (front < rear) {
    // Dequeue a vertex and explore its adjacent nodes
    int current = queue[++front];
    // Explore all adjacent vertices of the current vertex
    for (i = 1; i <= n; i++) {
      if (adj[current][i] && !visited[i]) {
        queue[++rear] = i;  // Enqueue the unvisited adjacent vertex
        visited[i] = 1;     // Mark it as visited
      }
    }
  }
}
void main() {
```

```
int v;
clrscr();
cout << "Enter the number of vertices: ";
cin >> n;
// Initialize visited array and queue
for (i = 1; i <= n; i++) {
   visited[i] = 0;
   queue[i] = 0;
}
cout << "Enter graph data in matrix form: \n";
for (i = 1; i <= n; i++) {
   for (j = 1; j <= n; j++) {
      cin >> adj[i][j];
   }
}
cout << "Enter the starting vertex: ";
cin >> v;
bfs(v);
cout << "The nodes which are reachable are: \n";
int allVisited = 1;
for (i = 1; i <= n; i++) {
   if (visited[i]) {
      cout << i << "\t";
   } else {
      allVisited = 0;
   }
```

```
    }
    if (!allVisited) {
        cout << "\nBFS is not possible. Not all nodes are reachable. \n";
    }
    getch();
}
```

Output:

Enter the number of vertices: 5

Enter graph data in matrix form:

0 1 0 0 1

1 0 1 0 0

1 1 0 1 0

0 0 1 0 0

1 0 0 0 0

Enter the starting vertex: 1

The node which are reachable are:

1 2 3 4 5

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 19

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for obtaining shortest path using Dijkstra Algorithm.

```
#include<iostream.h>

#include<conio.h>

#define INF 99999

#define V 9

void dijkstra(int graph[V][V], int src) {

  int dist[V];

  int visited[V];

    // Initialize distances and visited array

  for (int i = 0; i < V; i++) {

    dist[i] = INF;

    visited[i] = 0;

  }

  dist[src] = 0;

    for (int count = 0; count < V - 1; count++) {

    int u = -1;

    int min_dist = INF;

        // Find unvisited vertex with minimum distance

    for (int v = 0; v < V; v++) {

      if (!visited[v] && dist[v] < min_dist) {

        min_dist = dist[v];
```

```
            u = v;
          }
        }
        visited[u] = 1;
        // Update distances to adjacent vertices
        for (int v = 0; v < V; v++) {
          if (!visited[v] && graph[u][v] && dist[u] != INF &&
            dist[u] + graph[u][v] < dist[v]) {
            dist[v] = dist[u] + graph[u][v];
          }
        }
      }
      // Print the shortest distances
      cout << "Vertex \tDistance from Source \n";
      for (int i = 0; i < V; i++) {
        cout << i << " \t";
        if (dist[i] == INF)
          cout << "INF" << endl;
        else
          cout << dist[i] << endl;
      }
    }

    void main() {
      clrscr();
       int graph[V][V] = {
```

```
        {0, 6, 0, 0, 0, 0, 0, 8, 0},
        {6, 0, 8, 0, 0, 0, 0, 13, 0},
        {0, 8, 0, 7, 0, 6, 0, 0, 2},
        {0, 0, 7, 0, 9, 14, 0, 0, 0},
        {0, 0, 0, 9, 0, 10, 0, 0, 0},
        {0, 0, 6, 14, 10, 0, 2, 0, 0},
        {0, 0, 0, 0, 0, 2, 0, 1, 6},
        {8, 13, 0, 0, 0, 0, 1, 0, 7},
        {0, 0, 2, 0, 0, 0, 6, 7, 0}
    };
    dijkstra(graph, 0)
    getch();
}   Output:
                Vertex   Distance from Source
0 0
1 6
2 14
3 13
4 21
5 20
6 22
7 8
8 12
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 20

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in c for obtaining shortest Path using Floyd -Warshall Algorithm.

```
#include <iostream.h>

#include <conio.h>

#define V 4

#define INF 9999

int i,j;

void printSolution(int dist[V][V]) {

   cout << "\nShortest distances between every pair of vertices:\n";

   for ( i = 0; i < V; i++) {

         for ( j = 0; j < V; j++) {

            if (dist[i][j] == INF)

                  cout << "INF ";

            else

                  cout << dist[i][j] << "   ";

         }

         cout << "\n";

   }

}

void floydWarshall(int graph[V][V]) {

   int dist[V][V];

   for ( i = 0; i < V; i++)
```

```
            for ( j = 0; j < V; j++)
               dist[i][j] = graph[i][j];
   for (int k = 0; k < V; k++)
            for ( i = 0; i < V; i++)
               for ( j = 0; j < V; j++)
                     if (dist[i][k] + dist[k][j] < dist[i][j])
                        dist[i][j] = dist[i][k] + dist[k][j];
   printSolution(dist);
}
void main() {
   clrscr();
   int graph[V][V] = {
            {0, 5, INF, 10},
            {INF, 0, 3, INF},
            {INF, INF, 0, 1},
            {INF, INF, INF, 0}
   };
   floydWarshall(graph);
   getch();
}
```

Output:

The following matrix shows the shortest distances between every pair of vertices

```
  0     5    INF    10
 INF    0     3    INF
 INF   INF    0     1
       INF   INF   INF    0
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 21

Date of Performance: Date of Completion:

Name : Roll No :

Title// Program to implement Minimum Spanning Tree using Kruskal Algorithm

```
// Compatible with Turbo C++
#include <iostream.h>
#include <conio.h>
struct Edge {
   int u, v, weight;
};
class DSU {
public:
   int parent[10], rank[10];
   DSU(int n) {
      for (int i = 0; i < n; i++) {
         parent[i] = i;
         rank[i] = 0;
      }
   }
   int find(int u) {
      if (u != parent[u])
         parent[u] = find(parent[u]);
      return parent[u];
   }
```

```
    void unionSet(int u, int v) {
        int rootU = find(u);
        int rootV = find(v);
        if (rootU != rootV) {
            if (rank[rootU] > rank[rootV])
                parent[rootV] = rootU;
            else if (rank[rootU] < rank[rootV])
                parent[rootU] = rootV;
            else {
                parent[rootV] = rootU;
                rank[rootU]++;
            }
        }
    }
};
// Bubble Sort edges by weight
void bubbleSort(Edge edges[], int e) {
    for (int i = 0; i < e - 1; i++) {
        for (int j = 0; j < e - i - 1; j++) {
            if (edges[j].weight > edges[j + 1].weight) {
                Edge temp = edges[j];
                edges[j] = edges[j + 1];
                edges[j + 1] = temp;
            }
        }
    }
```

```
}
// Kruskal Algorithm
int kruskal(Edge edges[], int n, int e) {
   bubbleSort(edges, e);
   DSU dsu(n);
   int mstWeight = 0;
   cout << "\nEdges in the Minimum Spanning Tree:\n";
   for (int i = 0; i < e; i++) {
      if (dsu.find(edges[i].u) != dsu.find(edges[i].v)) {
         dsu.unionSet(edges[i].u, edges[i].v);
         cout << edges[i].u << " - " << edges[i].v
            << " : " << edges[i].weight << endl;
         mstWeight += edges[i].weight;
      }
   }
   return mstWeight;
}
void main() {
   clrscr();
   int n, e;
   Edge edges[10];
   cout << "Enter number of vertices: ";
   cin >> n;
   cout << "Enter number of edges: ";
   cin >> e;
   cout << "Enter edges (u v weight):\n";
```

```
    for (int i = 0; i < e; i++) {
        cin >> edges[i].u >> edges[i].v >> edges[i].weight;
    }
    int mstWeight = kruskal(edges, n, e);
    cout << "\nTotal weight of Minimum Spanning Tree: "<< mstWeight;
    getch();
}
```

Output:

```
Enter number of vertices: 4
Enter number of edges: 5
Enter edges (u v weight):
0 1 10
0 2 6
0 3 5
1 3 15
2 3 4
Edges in the Minimum Spanning Tree:
2 - 3 : 4
0 - 3 : 5
0 - 1 : 10
Total weight of Minimum Spanning Tree: 19
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 22

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement Minimum Spanning Tree using Prim's Algorithm

```
#include <iostream.h>
#include <conio.h>
void main() {
  clrscr();
  int n, i, j, ne = 1;
  int min, mincost = 0;
  int a = 0, b = 0;
  int cost[10][10];
  int visited[10];
  cout << "\nEnter the number of vertices: ";
  cin >> n;
  cout << "\nEnter the adjacency matrix (use 0 for no edge): \n";
  for (i = 1; i <= n; i++) {
    for (j = 1; j <= n; j++) {
      cin >> cost[i][j];
      if (cost[i][j] == 0)
        cost[i][j] = 999;   // Infinity
    }
    visited[i] = 0;
  }
  visited[1] = 1;   // Start from vertex 1
```

```
  cout << endl;
  while (ne < n) {
    min = 999;
    for (i = 1; i <= n; i++) {
      if (visited[i] == 1) {
        for (j = 1; j <= n; j++) {
          if (visited[j] == 0 && cost[i][j] < min) {
            min = cost[i][j];
            a = i;
            b = j;
          }
        }
      }
    }
    cout << "Edge (" << a << ", " << b << ") = " << min << endl;
    visited[b] = 1;
    mincost = mincost + min;
    ne++;
  }
  cout << "\nMinimum Spanning Tree total weight = " << mincost;
  getch();
}
```

output

Enter the number of vertices: 4

Enter the adjacency matrix (use 0 for no edge):

0 10 999 30

10 0 50 999

999 50 0 20

30 999 20 0

Edge (1, 2) = 10

Edge (1, 4) = 30

Edge (3, 4) = 20

Minimum Spanning Tree total weight = 60

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 23

Date of Performance: Date of Completion:

Name: Roll no:

Title:To implement a program in cpp for Hash Table.

```
#include <iostream.h>
#include <conio.h>
class HashTable
{
private:
  int table[10][10];   // 2D array for chaining
  int sizes[10];       // Size of each bucket
  int hashFunction(int key) {
    return key % 10;
  }
public:
  HashTable() {
    for (int i = 0; i < 10 ; i++) {
      sizes[i] = 0;
      for (int j = 0; j < 10; j++)
        table[i][j] = -1;
    }
  }
  void insert(int key) {
    int index = hashFunction(key);
    table[index][sizes[index]] = key;
```

```
    sizes[index]++;
}
void remove(int key) {
    int index = hashFunction(key);
    for (int i = 0; i < sizes[index]; i++) {
        if (table[index][i] == key) {
            for (int j = i; j < sizes[index] - 1; j++)
                table[index][j] = table[index][j + 1];
            table[index][sizes[index] - 1] = -1;
            sizes[index]--;
            break;
        }
    }
}
int search(int key) {
    int index = hashFunction(key);
    for (int i = 0; i < sizes[index]; i++) {
        if (table[index][i] == key)
            return 1;
    }
    return 0;
}
void display() {
    for (int i = 0; i < 10; i++) {
        cout << "Bucket " << i << ": ";
        for (int j = 0; j < sizes[i]; j++) {
```

```
            cout << table[i][j] << " -> ";
        }
        cout << "null" << endl;
    } }  };
void main() {
    clrscr();
    HashTable ht;
    ht.insert(10);
    ht.insert(20);
    ht.insert(15);
    ht.insert(7);
    ht.insert(25);
    cout << "Hash Table: " << endl;
    ht.display();
    int key = 15;
    cout << "\nSearching for key " << key << ": ";
    if (ht.search(key))
        cout << "Found" << endl;
    else
        cout << "Not Found" << endl;
    key = 20;
    cout << "\nRemoving key " << key << endl;
    ht.remove(key);
    cout << "\nUpdated Hash Table: " << endl;
    ht.display();
    getch();
```

}

Output:

Hash Table:

Bucket 0: null

Bucket 1: null

Bucket 2: null

Bucket 3: null

Bucket 4: null

Bucket 5: null

Bucket 6: null

Bucket 7: 7 -> null

Bucket 8: null

Bucket 9: 10 -> 20 -> 15 -> 25 -> null

Searching for key 15: Found

Removing key 20

Updated Hash Table:

Bucket 0: null

Bucket 1: null

Bucket 2: null

Bucket 3: null

Bucket 4: null

Bucket 5: null

Bucket 6: null

Bucket 7: 7 -> null

Bucket 8: null

Bucket 9: 10 -> 15 -> 25 -> null

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 24

Date of Performance: Date of Completion:

Name : Roll No :

Title: To implement a program in cpp for Linear Search using array.

```
#include <iostream.h>
#include <conio.h>
int linearSearch(int arr[], int n, int target) {
   for (int i = 0; i < n; i++) {
      if (arr[i] == target)
         return i;   // element found
   }
   return -1;         // element not found
}
void main() {
   clrscr();
   int arr[20], n, target, i, result;
   cout << "How many elements u want to enter: ";
   cin >> n;
   cout << "Enter array elements:\n";
   for (i = 0; i < n; i++) {
      cin >> arr[i];
   }
   cout << "Enter element to search: ";
   cin >> target;
   result = linearSearch(arr, n, target);
```

```
    if (result != -1)
        cout << "Element found at index " << result;
    else
        cout << "Element not found in the array";
    getch();
}
```

Output:

```
How  many  elements  u want  to enter:
5
Enter  array  elements:
10
20
30
40
50
Enter  element  to search:  30
Element found at index 2
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 25

Date of Performance: Date of Completion:

Name : Roll No :

Title:To implement a program in cpp for Binary Search using array.

```
#include <iostream.h>
#include <conio.h>
int binarySearch(int arr[], int n, int target) {
   int left = 0, right = n - 1, mid;
   while (left <= right) {
      mid = (left + right) / 2;
      if (arr[mid] == target)
         return mid + 1;   // +1 for location (not index)
      else if (arr[mid] < target)
         left = mid + 1;
      else
         right = mid - 1;
   }
   return -1;
}
void main() {
   clrscr();

   int arr[20], n, i, target, result;
   cout << "Enter number of elements: ";
   cin >> n;
```

```
   cout << "Enter " << n << " integers:\n";
   for (i = 0; i < n; i++) {
      cin >> arr[i];
   }
   cout << "Enter value to find: ";
   cin >> target;
   result = binarySearch(arr, n, target);
   if (result != -1)
      cout << target << " found at location " << result;
   else
      cout << "Element not found";
   getch();
}Output:
Enter  number  of elements:
5
Enter  5 integers:
10
20
30
40
50
Enter  value  to find:  50
50 found at location 5
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 26

Date of Performance: Date of Completion:

Name : Roll no:

Title:To implement a program in cpp for Bubble Sort.

```
#include<iostream.h>
#include<conio.h>
void main() {
  clrscr();
  int array[100], n, i, j, swap;
   cout << "Enter number of elements: ";
  cin >> n;
    cout << "\nEnter " << n << " Numbers: \n";
  for (i = 0; i < n; i++)
    cin >> array[i];
  // Bubble Sort
  for (i = 0; i < n - 1; i++) {
    for (j = 0; j < n - i - 1; j++) {
      if (array[j] > array[j + 1]) {
        swap = array[j];
        array[j] = array[j + 1];
        array[j + 1] = swap;
      }
    }
  }
   cout << "\nSorted Array: \n";
```

```
   for (i = 0; i < n; i++)
      cout << array[i] << endl;
    getch();
}
```

Output:

Enter number of elements: 5

Enter 5 Numbers:

42

25

33

17

8

Sorted Array:

8

17

25

33

42

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 27

Date of Performance: Date of Completion:

Name: Roll no:

Title:To implement a program in c pp for Selection Sort.

```
#include<iostream.h>
#include<conio.h>
void swap(int *a, int *b) {
   int temp = *a;
   *a = *b;
   *b = temp;
}
void selectionSort(int arr[], int size) {
   int i, j;
   for (i = 0; i < size; i++) {
      for (j = i; j < size; j++) {
         if (arr[i] > arr[j])
            swap(&arr[i], &arr[j]);
      }
   }
}
void main() {
   clrscr();
   int array[10], size;
   cout << "How many numbers you want to sort: ";
   cin >> size;
```

```
   cout << "\nEnter " << size << " numbers: \n";
   for (int i = 0; i < size; i++)
      cin >> array[i];

   selectionSort(array, size);

   cout << "\nSorted array is: \n";
   for (i = 0; i < size; i++)
      cout << array[i] << endl;
   getch();
}
```

Output:

```
How many numbers you want to sort: 5
Enter 5 numbers:
42
17
33
8
25
Sorted array is:
8
17
25
33
42
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 28

Date of Performance: Date of Completion:

Name : Roll no:

Title: To implement a program in c pp for Insertion Sort.

```
 #include<iostream.h>
#include<conio.h>
void main() {
   clrscr();
   int n, i, j, temp;
   int arr[64];
   cout << "Enter number of elements: ";
   cin >> n;
   cout << "\nEnter " << n << " integers: \n";
   for (i = 0; i < n; i++) {
      cin >> arr[i];
   }
   // Insertion Sort Algorithm
   for (i = 1; i < n; i++) {
      j = i;
      while (j > 0 && arr[j - 1] > arr[j]) {
         temp = arr[j];
         arr[j] = arr[j - 1];
         arr[j - 1] = temp;
         j--;
      }
```

```
    }
    cout << "\nSorted list in ascending order: \n";
    for (i = 0; i < n; i++) {
        cout << arr[i] << endl;
    }
    getch();
}
```

Output:

```
 Enter number of elements: 5
Enter 5 integers:
42
17
33
8
25
Sorted list in ascending order:
8
17
25
33
42
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 29

Date of Performance: Date of Completion:

Title: To implement a program in cpp for Radix Sort.

```
#include <iostream.h>
#include <conio.h>

int maximum(int arr[], int size)
{
   int max = arr[0];
   for (int i = 1; i < size; i++)
   {
      if (arr[i] > max)
         max = arr[i];
   }
   return max;
}
void countSort(int arr[], int size, int exponent)
{
   int output[100];
   int count[10] = {0};
   int i;
   // Store count of occurrences
   for (i = 0; i < size; i++)
      count[(arr[i] / exponent) % 10]++;
```

```
    // Change count[i] so that it contains actual position
    for (i = 1; i < 10; i++)
        count[i] += count[i - 1];
    // Build output array
    for (i = size - 1; i >= 0; i--)
    {
        output[count[(arr[i] / exponent) % 10] - 1] = arr[i];
        count[(arr[i] / exponent) % 10]--;
    }

    // Copy output to arr[]
    for (i = 0; i < size; i++)
        arr[i] = output[i];
}

void radixSort(int arr[], int size)
{
    int maxVal = maximum(arr, size);

    for (int exp = 1; maxVal / exp > 0; exp *= 10)
        countSort(arr, size, exp);
}

void main()
{
    clrscr();
```

```
    int arr[100], size, i;

    cout << "Enter the array size: ";
    cin >> size;

    cout << "Enter the array elements:\n";
    for (i = 0; i < size; i++)
    {
        cout << "arr[" << i << "] = ";
        cin >> arr[i];
    }

    cout << "\nArray before sorting:\n";
    for (i = 0; i < size; i++)
        cout << arr[i] << " ";
    radixSort(arr, size);
    cout << "\n\nArray after sorting:\n";
    for (i = 0; i < size; i++)
        cout << arr[i] << " ";

    getch();
}
```

Output:

Enter the array size: 6

Enter the array elements:

arr[0] = 170

arr[1] = 45

arr[2] = 75

arr[3] = 90

arr[4] = 802

arr[5] = 24

Array before sorting:

170 45 75 90 802 24

Array after sorting:

24 45 75 90 170 802

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 30

Date of Performance: Date of Completion:

Name: Roll no:

Title : To implement a program in cpp for Quick Sort.

```
#include <iostream.h>
#include <conio.h>
// Function to swap two elements
void swap(int &a, int &b)
{
   int temp = a;
   a = b;
   b = temp;
}
// Function to partition the array
int partition(int arr[], int low, int high)
{
   int pivot = arr[high];   // Pivot element
   int i = low - 1;
   for (int j = low; j < high; j++)
   {
      if (arr[j] <= pivot)
      {
         i++;
         swap(arr[i], arr[j]);
      }
```

```
    }
    swap(arr[i + 1], arr[high]);
    return i + 1;
}
// Quick Sort function
void quickSort(int arr[], int low, int high)
{
    if (low < high)
    {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
void main()
{
    clrscr();
    int arr[100], n, i;
    cout << "Enter the number of elements in the array: ";
    cin >> n;
    cout << "Enter the elements of the array:\n";
    for (i = 0; i < n; i++)
    {
        cout << "arr[" << i << "]: ";
        cin >> arr[i];
    }
```

```
    quickSort(arr, 0, n - 1);
    cout << "\nThe sorted array is:\n";
    for (i = 0; i < n; i++)
        cout << arr[i] << endl;
    getch();
}
```

Output:

```
 Enter the number of elements in the array: 6
Enter the elements of the array:
arr[0]: 42
arr[1]: 17
arr[2]: 33
arr[3]: 8
arr[4]: 25
arr[5]: 90

The sorted array is:
8
17
25
33
42
90
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 31

Date of Performance: Date of Completion:

Name: Roll no:

Title:To implement a program in cpp for Merge Sort.

```
#include <iostream.h>
#include <conio.h>
// Function to merge two subarrays
void merge(int arr[], int left, int mid, int right)
{
   int i, j, k;
   int n1 = mid - left + 1;
   int n2 = right - mid;
   int L[100], R[100];
   // Copy data to temporary arrays
   for (i = 0; i < n1; i++)
      L[i] = arr[left + i];
   for (j = 0; j < n2; j++)
      R[j] = arr[mid + 1 + j];
   i = 0;
   j = 0;
   k = left;
   // Merge the temp arrays back into arr[]
   while (i < n1 && j < n2)
   {
      if (L[i] <= R[j])
```

```
            arr[k++] = L[i++];
        else
            arr[k++] = R[j++];
    }
    // Copy remaining elements of L[]
    while (i < n1)
        arr[k++] = L[i++];
    // Copy remaining elements of R[]
    while (j < n2)
        arr[k++] = R[j++];
}
// Merge Sort function
void mergeSort(int arr[], int left, int right)
{
    if (left < right)
    {
        int mid = (left + right) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr, left, mid, right);
    }
}
void main()
{
    clrscr();
    int arr[100], n, i;
```

```
   cout << "Enter the size of the array: ";
   cin >> n;
   cout << "Enter the elements of the array:\n";
   for (i = 0; i < n; i++)
      cin >> arr[i];
   mergeSort(arr, 0, n - 1);
   cout << "\nThe sorted array is:\n";
   for (i = 0; i < n; i++)
      cout << arr[i] << endl;
   getch();    }
```

Output:

```
 Enter the size: 6
Enter the elements of the array:
34
7
23
90
12
5
The sorted array is:
5
7
12
23
34
90
```

Shri Jaykumar Rawal Institute of Technology Dondaicha

Practical: 32

Date of Performance: Date of Completion:

Name: Roll no:

Title: To implement a program in cpp for Heap Sort.

```
#include <iostream.h>
#include <conio.h>
// Function to swap two numbers
void swap(int *a, int *b)
{
   int temp = *a;
   *a = *b;
   *b = temp;
}
// Function to heapify a subtree rooted at index i
void heapify(int arr[], int n, int i)
{
   int largest = i;
   int left = 2 * i + 1;
   int right = 2 * i + 2;
   if (left < n && arr[left] > arr[largest])
      largest = left;
   if (right < n && arr[right] > arr[largest])
      largest = right;
   if (largest != i)
   {
```

```
        swap(&arr[i], &arr[largest]);
        heapify(arr, n, largest);
    }
}
// Heap Sort function
void heapSort(int arr[], int n)
{
    int i;
    // Build Max Heap
    for (i = n / 2 - 1; i >= 0; i--)
        heapify(arr, n, i);
    // Extract elements from heap
    for (i = n - 1; i > 0; i--)
    {
        swap(&arr[0], &arr[i]);
        heapify(arr, i, 0);
    }
}
// Function to display array
void printArray(int arr[], int n)
{
    int i;
    for (i = 0Output:
int arr[] = {10, 3, 5, 16, 92, 12, 56, 43};
Array before sorting:
10 3 5 16 92 12 56 43
```

Array after sorting:

3 5 10 12 16 43 56 92